Vulnerability-Aware Instance Reweighting For Adversarial Training

Olukorede Fakorede*  
Department of Computer Science  
Iowa State University

Ashutosh Nirala  
Department of Computer Science  
Iowa State University

Modeste Atsague  
Department of Computer Science  
Iowa State University

Jin Tian  
Department of Computer Science  
Iowa State University

Reviewed on OpenReview: https://openreview.net/forum?id=kdPcLdJbt1

Abstract

Adversarial Training (AT) has been found to substantially improve the robustness of deep learning classifiers against adversarial attacks. AT involves obtaining robustness by including adversarial examples in training a classifier. Most variants of AT algorithms treat every training example equally. However, recent works have shown that better performance is achievable by treating them unequally. In addition, it has been observed that AT exerts an uneven influence on different classes in a training set and unfairly hurts examples corresponding to classes that are inherently harder to classify. Consequently, various reweighting schemes have been proposed that assign unequal weights to robust losses of individual examples in a training set. In this work, we propose a novel instance-wise reweighting scheme. It considers the vulnerability of each natural example and the resulting information loss on its adversarial counterpart occasioned by adversarial attacks. Through extensive experiments, we show that our proposed method significantly improves over existing reweighting schemes, especially against strong white and black-box attacks.

1 Introduction

The practical application of deep learning in safety-critical domains has been thrown into doubt following the observed brittleness of deep learning to well-crafted adversarial perturbations [Szegedy et al., 2013]. This chilling observation has led to an array of methods aimed at making deep learning classifiers robust to these adversarial perturbations. Prominent among these proposed methods is adversarial training [Goodfellow et al., 2015; Madry et al., 2018]. Adversarial training (AT) is an effective method that typically involves the introduction of adversarial examples in training a deep learning classifier. Several AT variants have been proposed yielding modest improvements [Wang et al., 2019; Ding et al., 2019; Kannan et al., 2018; Zhang et al., 2019]. More recently, methods have been proposed to boost the performance of the existing AT variants even further, including adversarial weight perturbation [Wu et al., 2020], utilizing hypersphere
embedding (Pang et al., 2020; Fakorede et al., 2023), and augmenting dataset with unlabeled and/or extra labeled data (Carmon et al., 2019; Alayrac et al., 2019; Zhai et al., 2019).

Despite the generally impressive performance of AT against adversarial attacks, Xu et al. (2021) raised concerns about fairness in AT. It is observed that AT encourages significant disparity in natural and robust accuracy among different classes. Furthermore, AT disproportionately hurts the robust accuracy of input examples that are intrinsically harder to classify by a naturally trained classifier. For example, a naturally trained PreactResNet-18 on the CIFAR-10 dataset classifies “cat” and “ship” at approximately 89% and 96% accuracy, respectively, while the robust accuracy of “cat” and “ship” produced by an adversarially trained PreactResNet-18 on PGD-attacked CIFAR-10 dataset are approximately 17% and 59% respectively (Xu et al., 2021). In other words, the gap between the standard accuracy and robust accuracy for “cat” is 72%, whereas the difference between the standard and robust accuracy for “ship” is 37%. This disparity suggests that adversarial examples corresponding to certain classes may be treated differently by AT.

Adversarial examples are crafted from natural examples that exhibit varying degrees of intrinsic vulnerability measured by their closeness to the class decision boundaries. Intuitively, adversarial examples corresponding to intrinsically vulnerable examples are moved farther across the decision boundary into wrong classes which makes them easy to misclassify. Zhang et al. (2020) observed that AT encourages learning adversarial variants of less vulnerable natural examples at the expense of the intrinsically susceptible ones as the training proceeds. This may partly explain the phenomenon of robust overfitting (Chen et al., 2020; Zhang et al., 2020). Therefore, the performance of AT may be improved by assigning higher weights to the robust losses of adversarial variants of vulnerable natural examples.

The idea of reweighting robust losses of adversarial examples has recently been explored in the literature. GAIRAT (Zhang et al., 2020) assigns smaller or larger weights to the robust losses of adversarial examples based on the geometric distance of their corresponding natural counterpart to the decision boundary. Specifically, GAIRAT measures the geometric distance using the least number of PGD steps needed to misclassify the natural example. As a result, the GAIRAT reweighting function can only take a few values (corresponding to discrete PGD steps) and is unstable because its value depends largely on the initial starting point of each natural example which may change depending on the attack path (Liu et al., 2021). Liu et al. (2021) proposed reweighting robust losses based on the margin between the estimated ground-truth probability of the adversarial example and the probability of the most confusing label. We contend that this method does not exploit information about the intrinsic vulnerability of a natural example. More importantly, we observe that these methods only yield competitive robustness on attacks like PGD and FGSM, but underperform against stronger white-box or black-box attacks CW, AA, or SPSA.

This paper proposes a novel instance-wise weight assignment function for assigning importance to the robust losses of adversarial examples used for adversarial training. Our weight assignment function considers the intrinsic vulnerability of individual natural examples from which adversarial examples used during training are crafted. We capture the intrinsic vulnerability of each natural example using the likelihood of it being correctly classified, which we estimate using the model’s confidence about the example belonging to its true class. In addition, we argue that adversarial attacks have a unique impact on each individual example. This contributes to the disparity in robustness accuracy exhibited by examples of different classes. Hence, we compute the discrepancies between a model’s prediction on each natural example and its corresponding adversarial example as another measure for the vulnerability of the example.

We summarize the contributions of this paper as follows:

1. We propose a novel Vulnerability-aware Instance Reweighting (VIR) function for adversarial training. The proposed reweighting function takes consideration of the intrinsic vulnerability of individual examples used for adversarial training and the information loss occasioned by adversarial attacks on natural examples.

2. We experimentally demonstrate the effectiveness of the proposed reweighting strategy in improving adversarial training.

3. We show that existing reweighting methods GAIRAT (Zhang et al., 2020) and MAIL (Liu et al., 2021) only yield significant robustness against attacks FGSM and PGD at the expense of stronger
attacks CW (Carlini & Wagner, 2017), AutoAttack (Croce & Hein, 2020b), and FMN (Pintor et al., 2021). Using various datasets and models, we show that the proposed VIR method consistently improves over the existing reweighting methods across various white-box and black-box attacks.

2 RELATED WORK

Adversarial Attacks. Since it became known that deep neural networks (DNN) are vulnerable to norm-bounded adversarial attacks (Biggio et al., 2013; Szegedy et al., 2013), a number of sophisticated adversarial attack algorithms have been proposed (Goodfellow et al., 2015; Madry et al., 2018; Carlini & Wagner, 2017). Adversarial attacks can broadly be classified into white-box and black-box attacks. White-box attacks are crafted with the attacker having full access to the model parameters. Prominent white-box attacks include Fast Gradient Sign Method (Goodfellow et al., 2015), DeepFool (Moosavi-Dezfooli et al., 2016), C&W (Carlini & Wagner, 2017), Projected Gradient Descent (PGD) (Madry et al., 2018) etc. On the other hand, in black-box settings, the attacker has no direct access to the model parameters, and black-box attacks usually rely on a substitute model (Papernot et al., 2017), or gradient estimation of the target model (Hyas et al., 2018; Esato et al., 2018).

Adversarial Robustness. To mitigate the potential threat of adversarial attacks, extensive research has been conducted, leading to various methods (Guo et al., 2018; Song et al., 2017; Papernot et al., 2016; Madry et al., 2018; Zhang et al., 2019; Wang et al., 2019; Ding et al., 2019; Kannan et al., 2018), to cite a few. Adversarial Training (AT) (Goodfellow et al., 2015; Madry et al., 2018) involves training a classifier with adversarial examples, has been found to be ineffective against strong attacks (Athalye et al., 2018). Adversarial Training (AT) (Goodfellow et al., 2015) etc. On the other hand, in black-box settings, the attacker has no direct access to the model parameters, and black-box attacks usually rely on a substitute model (Papernot et al., 2017), or gradient estimation of the target model (Hyas et al., 2018; Esato et al., 2018).

Re-weighting. Recent works by Zhang et al. (2020) and Liu et al. (2021) have argued for assigning different weights to the losses corresponding to different adversarial examples in the training set. Zhang et al. (2020) assigns weights to an example based on its distance to the decision boundary. Examples that are closer to the decision boundary are assigned larger weights as follows:

\[ w(x_i, y_i) = \frac{1 + \tanh(\lambda + 5 \times (1 - 2k(x_i, y_i)/K))}{2} \tag{2} \]

where \( k(x_i, y_i) \) is the least number of PGD steps to cause misclassification of \( x_i \), \( K = 10 \) is the number of PGD steps used for generating the attack, and \( \lambda \) is set to -1. The assignment function in Eq. (2) is used to reweight the robust losses on each adversarial example in Eq. (1).

Unlike Zhang et al. (2020) which uses a re-weighting function that is discrete (i.e. the weighting function depends on \( k \) PGD iterations) and path-dependent, Liu et al. (2021) proposed a re-weighting scheme that is continuous and path-independent based on the probability margin between the estimated ground-truth probability of an adversarial example and the probability of the class closest to the ground-truth as follows:

\[ PM(x, y; \theta) = f_\theta(x’)_y - \max_{j \neq y} f_\theta(x’)_j \tag{3} \]

The weight assignment function is then defined as:

\[ w(x_i, y_i) = \text{sigmoid}(-\gamma(PM_i - \beta)) \tag{4} \]
where $\gamma$ and $\beta$ are hyperparameters.

This work takes a different perspective on reweighting robust losses. We consider the inherent vulnerability of the natural examples used for crafting adversarial examples and the impact of adversarial attacks on each natural example.

3  PRELIMINARIES

We use bold letters to denote vectors. We denote $D = \{x_i, y_i\}_{i=1}^n$ as a data set of input feature vectors $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$ and labels $y_i \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ represent a feature space and a label set, respectively.

Let $f_\theta : \mathcal{X} \rightarrow \mathcal{R}^C$ denote a deep neural network (DNN) classifier parameterized by $\theta$ where $C$ represents the number of output classes. For any $x \in \mathcal{X}$, let the class label predicted by $f_\theta$ be $F_\theta(x) = \arg \max_k f_\theta(x)_k$, where $f_\theta(x)_k$ denotes the $k$-th component of $f_\theta(x)$. $f_\theta(x)_y$ is the likelihood of $x$ belonging to class $y$.

$\text{KL}(P \parallel Q)$ is used to represent the Kullback-Leibler divergence of distributions $P$ and $Q$.

We denote $\| \cdot \|_p$ as the $l_p$- norm over $\mathbb{R}^d$, that is, for a vector $x \in \mathbb{R}^d$, $\|x\|_p = \left( \sum_{i=1}^d |x_i|^p \right)^{1/p}$. An $\epsilon$-neighborhood for $x$ is defined as $B_\epsilon(x) : \{x' \in \mathcal{X} : \|x' - x\|_p \leq \epsilon \}$. An adversarial example corresponding to a natural input $x$ is denoted as $x'$. We refer to a DNN trained only on natural examples as standard network, and the one trained using adversarial examples as robust network or adversarially trained network.

4  PROPOSED METHOD

The proposed approach is motivated by two major factors: (1) the unfairness exhibited by adversarial training - adversarial examples crafted from intrinsically vulnerable natural examples are underrepresented in the later stages of adversarial training (Zhang et al., 2020), (2) the discrepancy between individual natural examples and their corresponding attacked variants.

We propose an effective reweighting strategy for assigning importance to robust losses based on these two factors. We will make a case for how the information about the inherent vulnerability of natural examples and the effect of adversarial attacks on these examples may be captured and utilized as inputs to the proposed weight assignment function.

4.1 Vulnerability of Natural Examples

Based on the findings in (Xu et al., 2021; Ma et al., 2022), supported by the confusion matrices in Figure 1, it is clear that adversarial training hurts adversarial examples of certain classes more than others, especially adversarial examples crafted from natural examples which are harder to classify under natural training. These natural examples are characterized by their closeness to the class decision boundaries (Xu et al., 2021; Ma et al., 2022).
Given a natural input \( x \) of a natural example in terms of a model’s estimated class probability, it makes models less confident in distinguishing their examples. Given this insight, we define the vulnerability that contains certain similar features. We believe that this relative feature similarity in semantically closer classes makes models less confident in distinguishing their examples. Given this insight, we define the vulnerability of a natural example in terms of a model’s estimated class probability.

Given a natural input \( x \) with its corresponding true label \( y \), and a model \( f_\theta(\cdot) \), the model’s estimated probability of \( x \) belonging to \( y \) is given as \( f_\theta(x)_y \). Note that \( y \) is not necessarily the same as \( arg \max_k f_\theta(x)_k \). We formally define the notion of relative vulnerability of inputs below.

**Definition 4.1 (Relative vulnerability of examples.)** Given two input-label pairs \((x_1, y_1)\) and \((x_2, y_2)\), we say the pair \((x_1, y_1)\) is more vulnerable than \((x_2, y_2)\), if \( f_\theta(x_1)_{y_1} < f_\theta(x_2)_{y_2} \).

Based on Definition 4.1, model \( f_\theta(\cdot) \) is less confident in estimating the true class of more vulnerable examples. We provide more theoretical explanation to our notion of relative example vulnerability in Appendix A.

Given that adversarial training involves training on adversarial examples, and it unfairly hurts adversarial examples crafted from vulnerable examples, it is intuitive that adversarial training sets up its decision boundary to favor adversarial examples of invulnerable classes, as observed in (Xu et al., 2021). This is indicated by the relatively low robust accuracy recorded on certain classes. Given this insight, we consider assigning unequal weights to adversarial examples based on the relative vulnerability of their original (natural) examples with respect to their respective true labels.

Our proposed re-weighting strategy for adversarial training considers the vulnerability of each natural example used to generate adversarial examples for training. This vulnerability for an input \( x \) with label \( y \) is given by \( f_\theta(x)_y \). Adversarial examples corresponding to natural examples which are intrinsically vulnerable are assigned higher weights. We define a score function denoted as \( S_v(x, y) \) for adaptively assigning importance to input example \( x \) based on a model’s confidence that \( x \) belongs to the true label \( y \). The score function is as follows:

\[
S_v(x_i, y_i) = \alpha \cdot e^{-\gamma f_\theta(x_i)_y}, \quad (5)
\]

where \( x_i \) is a natural example, \( y_i \) is the true label of \( x_i \), \( e^{(-)} \) represents an exponential function, and \( \gamma \geq 1.0 \) is a real-valued hyperparameter. \( \alpha \) is introduced for numerical stability to ensure that \( S_v(x, y) \) values are not too small and are useful. The formulation in Eq. 5 ensures that higher values are returned for more vulnerable examples which have lower \( f_\theta(x_i)_y \) values. Higher \( \gamma \) values will result in a larger disparity in weights between vulnerable examples and less vulnerable examples.

### 4.2 Disparity between Natural and Adversarial Examples

The likelihood estimate of correctly classifying a natural example provides information about its intrinsic vulnerability before an adversarial perturbation is applied to it. However, it does not supply information about the discrepancies in the features learned by the model on natural examples and their corresponding adversarial variants, which largely explain the large disparity between natural and robust errors.

Ilyas et al. (2019) and Tsipras et al. (2018) characterized learned features into robust and non-robust. Robust features refer to features that remain correlated to the true label under adversarial perturbation. In contrast, non-robust features are highly predictive, however, they are brittle and can be anti-correlated with the true label under adversarial perturbations. Tsipras et al. (2018) showed that a DNN classifier is able to learn any useful features on natural examples, but learns only robust features on adversarial examples, assigning zero or infinitesimal weight values to the predictive non-robust features.
The depletion of features in the adversarial examples results in different information loss for different examples. We note that there may be a variation in non-robust features learned by a DNN classifier in the different adversarial examples used in training. For instance, consider two input-label pairs \((x_1, y_1), (x_2, y_2)\) and their corresponding perturbed variants \(x'_1, x'_2\). The robust features in \(x'_1\) may have a stronger correlation to \(y_1\) than the robust features in \(x'_2\) to \(y_2\). In fact, intrinsically vulnerable examples which are relatively weakly correlated with their classes may be affected more by an adversary since their features are further depleted. We propose to score the vulnerability of an example \(x\) as a weight for \(x\) by the discrepancy between the model’s output prediction on \(x\) and that on its adversarial variant \(x'\). For simplicity, we score this discrepancies using the KL-divergence as follows:

\[
S_d(x_i, x'_i) = KL(f_\theta(x_i)\|f_\theta(x'_i))
\]

where \(S_d\) denotes the proposed discrepancy score function, and \(f_\theta(x)\) and \(f_\theta(x')\) denote the model’s predictions on natural and the corresponding adversarial examples respectively.

We note that the KL-divergence between a model’s predictions on natural examples and adversarial examples is characterized as a boundary error in \textit{TRADES} \cite{Zhang2019}, and is utilized as a regularization term for improving robustness (see Eq. \ref{eq:4.4}). In contrast, here, it is used as a weighting (a value) and the gradient of the KL-divergence in Eq. \ref{eq:4.5} is not used.

### 4.3 Weight Assignment Function

We propose a weight assignment function for re-weighting robust losses based on the observations in Sections 4.1 and 4.2. The proposed \textit{Vulnerability-aware Instance Reweighting (VIR)} function for assigning importance to the example \(x_i\) in a training set is as follows:

\[
w(x_i, x'_i, y_i) = S_o(x_i, y_i) \cdot S_d(x_i, x'_i) + \beta
\]

where \(\beta\) is a hyperparameter. \(S_o(x, y)\) ensures that emphasis is given to vulnerable examples which are disproportionately hurt by adversarial training. The proposed weight assignment function assigns the largest weights to robust losses corresponding to vulnerable examples having the most significant discrepancy between the model’s outputs on the natural and adversarial variants. Furthermore, the least weights are assigned to robust losses corresponding to invulnerable examples having the least discrepancy between the model’s outputs on them and their corresponding adversarial variants. The hyperparameter \(\beta\) allows us to lower-bound the reweighting function and to balance the relative strength of the lowest and highest weight values.

As a comparison, \textit{GAIRAT} \cite{Zhang2019} assigns importance to robust losses based on the \(k\) PGD steps required to attack an example \(x\). The intuition is that vulnerable inputs take fewer steps to move across the decision boundary. However, as noted in \cite{Liu2021}, this approach is problematic because it is path-dependent, i.e., two inputs with similar starting points may reach their end points using different paths, thus making it unreliable. In addition, the reweighting function is constrained to accepting a few discrete values. In contrast, the proposed reweighting function in eqn. \ref{eq:4.6} takes continuous values and is path-independent.

### 4.4 Applying the Weight Assignment Function

We apply the proposed weight assignment function to prominent adversarial training methods vanilla AT \cite{Madry2018} and \textit{TRADES} \cite{Zhang2019} by assigning importance to robust losses computed by these adversarial training methods. Specifically, we re-write the training objective of the vanilla AT as:

\[
\sum_i w(x_i, x'_i, y_i) \cdot L_{CE}(f_\theta(x'_i), y_i)
\]

where \(L_{CE}\) refers to the cross-entropy loss function. The TRADES robust loss, originally stated as:

\[
\sum_i L_{CE}(f_\theta(x_i), y) + \frac{1}{\lambda} \cdot KL(f_\theta(x_i)\|f_\theta(x'_i)),
\]
is re-written as:
\[
\sum_i L_{CE}(f_0(x_i), y) + \frac{1}{\lambda} \cdot w(x_i, x'_i, y_i) \cdot KL(f_0(x_i) \parallel f_0(x'_i))
\]  
(10)

where \(\lambda\) is a regularization hyper-parameter. We term the training objectives in Eq. (8) and (10) as VIR-AT and VIR-TRADES respectively.

**Burn-in Period.** During the training, the weight \(w(x_i, x'_i, y_i)\) is set to 1 in the initial epochs. The application of the proposed weight assignment function is delayed to later epochs. This is because at the initial training phase, the deep model has not sufficiently learned, and thus is less informative. Disregarding this fact may mislead the training process. Zhang et al. (2020) used a similar approach in implementing their re-weighting strategy.

Our proposed VIR-AT algorithm is summarized in the following:

**Algorithm 1 VIR-AT Algorithm.**

```
Input: a neural network model with the parameters \(\theta\), step sizes \(\kappa_1\) and \(\kappa_2\), and a training dataset \(\mathcal{D}\) of size \(n\).
Output: a robust model with parameters \(\theta^*\)
1: for epoch = 1 to num_epochs do
   2:     for batch = 1 to num_batches do
   3:         sample a mini-batch \(\{(x_i, y_i)\}_{i=1}^M\) from \(\mathcal{D}\); \(\triangleright\) mini-batch of size \(M\).
   4:     for \(i = 1\) to \(M\) do
   5:         \(x'_i \leftarrow x_i + 0.001 \cdot N(0, 1)\), where \(N(0, I)\) is the Gaussian distribution with zero mean and
   6:         identity variance.
   7:   for \(k = 1\) to \(K\) do
   8:       \(x'_i \leftarrow \Pi_{B_i(x_i)} (x_i + \kappa_1 \cdot \text{sign}(\nabla_{x'_i} \cdot L(f_0(x'_i), y_i))); \) \(\triangleright\) \(\Pi\) is a projection operator.
   9:   end for
10: \(S_0(x_i, y_i) \leftarrow \alpha \cdot e^{-\gamma f_0(x_i)}\)
11: \(S_d(x'_i, x_i) \leftarrow KL(f_0(x'_i) \parallel f_0(x_i))\)
12: \(w_i(x_i, x'_i, y_i) \leftarrow S_0(x_i, y_i) \cdot S_d(x'_i, x_i) + \beta; \) \(\triangleright\) \(w_i(x_i, x'_i, y_i) \leftarrow 1\) if epoch \(\leq 76\)
13: end for
14: \(\theta \leftarrow \theta - \kappa_2 \nabla_{\theta} \sum_{i=1}^M w_i(x'_i, x_i, y_i) \cdot L(f_0(x'_i), y_i)\)
15: end for
```

5 EXPERIMENTAL SECTION

In this section, we verify the effectiveness of the proposed re-weighting function through extensive experiments on various datasets including CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), and TinyImageNet (Deng et al., 2009). We employed ResNet-18 (RN-18) (He et al., 2016) and WideResNet-34-10 (WRN-34-10) (He et al., 2016) as the backbone models for exploring the effectiveness of the proposed method on CIFAR-10, while CIFAR-100, SVHN, and TinyImageNet are evaluated on ResNet-18.

5.1 Experimental Settings

The models are trained for 115 epochs, using mini-batch gradient descent with momentum 0.9, batch size 128, weight decay 3.5e-3 (RN-18) and 7e-4 (WRN-34-10). The learning rates are set to 0.01 and 0.1 for RN-18 and WRN-34-10 respectively. In both cases, the learning rates are decayed by a factor of 10 at 75th, and then at 90th epoch. In VIR-AT and VIR-TRADES, we introduced the proposed reweighting function on the 76th epoch following Zhang et al. (2020).
The adversarial examples used during training are obtained by perturbing each image using the Projected Gradient Descent (PGD) [Madry et al., 2018] with the following hyperparameters: $l_\infty$ norm $\epsilon = 8/255$, step-size $\kappa = 2/255$, and $K = 10$ iterations.

5.2 Baselines

We compare the robustness obtained using VIR-AT and VIR-TRADES with prominent AT methods vanilla-AT [Madry et al., 2018], TRADES [Zhang et al., 2019], and MART [Wang et al., 2019]. In addition, we compare with the state-of-the-art reweighting schemes GAIRAT [Zhang et al., 2020] and MAIL [Liu et al., 2021]. We conducted additional experiments on recent data augmentation-based defense [Wang et al., 2023] and the results are provided in Appendix B.

5.3 Hyperparameters

**Baseline Hyperparameters.** The trade-off hyperparameter $\lambda$ is set to 6.0 for training WRN-34-10 and 4.0 for RN-18 with TRADES. As recommended by the authors, we set the regularization hyperparameter $\beta$ to 5.0 for training with MART.

**VIR Hyperparameters.** The values of constants $\alpha$ and $\beta$ are heuristically determined and set to 7.0 and 0.007 respectively in VIR-AT and 8.0 and 1.6 in VIR-TRADES. Similarly, we set the value of $\gamma$ to 10.0 and 3.0 in VIR-AT and VIR-TRADES respectively. We set the value of $\gamma$ to 3.0 for training TinyImageNet with VIR-AT. Also, the value of $\lambda$ is set to 5.0 for training VIR-TRADES.

5.4 Threat Models

The performance of the proposed reweighting function was evaluated using attacks under White-box and Black-box settings and Auto attack.

**White-box attacks.** These attacks have unfettered access to model parameters. To evaluate robustness on CIFAR-10 using RN-18 and WRN-34-10, we apply the Fast Gradient Sign Method (FGSM) [Goodfellow et al., 2015] with $\epsilon = 8/255$ ; PGD attack with $\epsilon = 8/255$, step size $\kappa = 1/255$, $K = 100$; CW (CW loss [Carlini & Wagner, 2017] optimized by PGD-20) attack with $\epsilon = 8/255$, step size 1/255. In addition, we evaluated the robustness of trained models against 100 iterations of $l_\infty$ version of Fast Minimum-norm (FMN) attacks [Pintor et al., 2021]. On SVHN and CIFAR-100, we apply PGD attack with $\epsilon = 8/255$, step size $\kappa = 1/255$, $K = 100$. We limited the white-box evaluation on TinyImageNet to PGD-20.

**Black-box attacks.** Under black-box settings, the adversary has no access to the model parameters. We evaluated robust models trained on CIFAR-10 against strong query-based black-box attacks Square [Andriushchenko et al., 2020] with 5,000 queries and SPSA [Uesato et al., 2018] with 100 iterations, perturbation size 0.001 (gradient estimation), learning rate = 0.01, and 256 samples for each gradient estimation. All black-box evaluations are made on trained WRN-34-10.

**Ensemble of Attacks.** Trained models are tested on powerful ensembles of attacks such as Autoattack [Croce & Hein, 2020b], which consisting of APGD-CE [Croce & Hein, 2020b], APGD-T [Croce & Hein, 2020b], FAB-T [Croce & Hein, 2020a], and Square (a black-box attack) [Andriushchenko et al., 2020] attacks. In addition, we evaluated the trained models on the Margin Decomposition Ensemble (MDE) attack [Ma et al., 2023].

5.5 Performance Evaluation

We summarize our results on CIFAR-10 using RN-18 and WRN-34-10 in Tables 1 and 2 respectively. Moreover, we report results on CIFAR-100, SVHN, and Tiny Imagenet using RN-18 in Tables 3 and 4. Finally, black-box evaluations are made on trained WRN-34-10, and the results are reported in Table 5. Experiments were repeated four times with different random seeds; the mean and standard deviation are subsequently calculated. Results are reported as mean ± std.
Table 1: Comparing white-box attack robustness (accuracy %) for ResNet-18 on CIFAR-10. For all methods, distance $\epsilon = 0.031$. We highlight the best-performing method under each attack.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>FGSM</th>
<th>PGD-100</th>
<th>CW</th>
<th>FMN-100</th>
<th>AA</th>
<th>MDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>84.12 ± 0.16</td>
<td>57.88 ± 0.13</td>
<td>51.58 ± 0.17</td>
<td>51.75 ± 0.23</td>
<td>48.92 ± 0.25</td>
<td>47.92 ± 0.35</td>
<td>47.90 ± 0.27</td>
</tr>
<tr>
<td>TRADES</td>
<td>83.56 ± 0.35</td>
<td>57.82 ± 0.32</td>
<td>52.07 ± 0.25</td>
<td>52.26 ± 0.07</td>
<td>49.74 ± 0.30</td>
<td>48.32 ± 0.19</td>
<td>48.29 ± 0.19</td>
</tr>
<tr>
<td>MART</td>
<td>80.32 ± 0.38</td>
<td>58.01 ± 0.19</td>
<td>54.03 ± 0.28</td>
<td>49.29 ± 0.11</td>
<td>49.82 ± 0.08</td>
<td>47.61 ± 0.27</td>
<td>47.55 ± 0.12</td>
</tr>
<tr>
<td>GAIRAT</td>
<td>83.33 ± 0.19</td>
<td>60.20 ± 0.20</td>
<td>54.91 ± 0.19</td>
<td>40.95 ± 0.30</td>
<td>38.62 ± 0.29</td>
<td>32.80 ± 0.33</td>
<td>32.70 ± 0.18</td>
</tr>
<tr>
<td>MAIL-AT</td>
<td>84.32 ± 0.46</td>
<td>60.11 ± 0.39</td>
<td>55.25 ± 0.23</td>
<td>48.88 ± 0.11</td>
<td>46.83 ± 0.32</td>
<td>44.22 ± 0.21</td>
<td>44.14 ± 0.21</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>82.93 ± 0.13</td>
<td>59.62 ± 0.08</td>
<td>54.36 ± 0.17</td>
<td>53.11 ± 0.17</td>
<td>51.95 ± 0.09</td>
<td>51.03 ± 0.16</td>
<td>50.89 ± 0.12</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>84.59 ± 0.18</td>
<td>61.35 ± 0.13</td>
<td>56.42 ± 0.18</td>
<td>52.18 ± 0.15</td>
<td>50.56 ± 0.12</td>
<td>48.21 ± 0.08</td>
<td>48.04 ± 0.08</td>
</tr>
</tbody>
</table>

Table 2: Comparing white-box attack robustness (accuracy %) for WideResNet-34-10 on CIFAR-10. For all methods, distance $\epsilon = 0.031$. The best-performing methods under each attack are highlighted.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>FGSM</th>
<th>PGD-100</th>
<th>CW</th>
<th>FMN-100</th>
<th>AA</th>
<th>MDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>86.17 ± 0.26</td>
<td>61.68 ± 0.13</td>
<td>54.45 ± 0.31</td>
<td>55.17 ± 0.33</td>
<td>54.09 ± 0.21</td>
<td>51.90 ± 0.28</td>
<td>51.74 ± 0.19</td>
</tr>
<tr>
<td>TRADES</td>
<td>85.20 ± 0.25</td>
<td>61.47 ± 0.35</td>
<td>54.81 ± 0.31</td>
<td>56.02 ± 0.29</td>
<td>53.95 ± 0.10</td>
<td>53.09 ± 0.18</td>
<td>52.71 ± 0.12</td>
</tr>
<tr>
<td>MART</td>
<td>84.59 ± 0.11</td>
<td>62.20 ± 0.14</td>
<td>56.45 ± 0.16</td>
<td>54.52 ± 0.11</td>
<td>53.17 ± 0.12</td>
<td>51.21 ± 0.23</td>
<td>50.92 ± 0.15</td>
</tr>
<tr>
<td>GAIRAT</td>
<td>85.24 ± 0.19</td>
<td>62.67 ± 0.36</td>
<td>57.09 ± 0.17</td>
<td>44.96 ± 0.2</td>
<td>44.50 ± 0.05</td>
<td>42.29 ± 0.11</td>
<td>41.92 ± 0.15</td>
</tr>
<tr>
<td>MAIL-AT</td>
<td>84.83 ± 0.39</td>
<td>64.09 ± 0.32</td>
<td>58.86 ± 0.25</td>
<td>51.26 ± 0.20</td>
<td>51.64 ± 0.15</td>
<td>47.10 ± 0.22</td>
<td>47.04 ± 0.11</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>84.95 ± 0.21</td>
<td>63.07 ± 0.17</td>
<td>57.56 ± 0.21</td>
<td>56.92 ± 0.19</td>
<td>55.72 ± 0.13</td>
<td>54.55 ± 0.26</td>
<td>54.14 ± 0.09</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>87.13 ± 0.36</td>
<td>64.71 ± 0.29</td>
<td>59.82 ± 0.29</td>
<td>56.11 ± 0.16</td>
<td>54.14 ± 0.23</td>
<td>51.94 ± 0.22</td>
<td>51.83 ± 0.12</td>
</tr>
</tbody>
</table>

Table 3: Comparing white-box attack robustness (accuracy %) for RN-18 on SVHN and CIFAR-100. For all methods, distance $\epsilon = 0.031$.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD-100</th>
<th>AA</th>
<th>Natural</th>
<th>PGD-100</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>92.94 ± 0.46</td>
<td>54.74 ± 0.28</td>
<td>45.94 ± 0.29</td>
<td>59.60 ± 0.03</td>
<td>28.57 ± 0.25</td>
<td>24.75 ± 0.21</td>
</tr>
<tr>
<td>TRADES</td>
<td>92.14 ± 0.41</td>
<td>55.24 ± 0.23</td>
<td>45.64 ± 0.29</td>
<td>60.73 ± 0.33</td>
<td>29.83 ± 0.25</td>
<td>24.83 ± 0.29</td>
</tr>
<tr>
<td>MART</td>
<td>91.84 ± 0.46</td>
<td>55.54 ± 0.21</td>
<td>43.39 ± 0.36</td>
<td>54.19 ± 0.26</td>
<td>29.94 ± 0.21</td>
<td>25.30 ± 0.56</td>
</tr>
<tr>
<td>GAIRAT</td>
<td>90.47 ± 0.58</td>
<td>61.37 ± 0.28</td>
<td>37.27 ± 0.31</td>
<td>58.43 ± 0.26</td>
<td>25.74 ± 0.41</td>
<td>17.57 ± 0.33</td>
</tr>
<tr>
<td>MAIL-AT</td>
<td>91.54 ± 0.51</td>
<td>62.16 ± 0.18</td>
<td>41.18 ± 0.29</td>
<td>60.74 ± 0.15</td>
<td>27.62 ± 0.27</td>
<td>22.44 ± 0.53</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>89.24 ± 0.23</td>
<td>58.63 ± 0.32</td>
<td>50.06 ± 0.22</td>
<td>59.20 ± 0.35</td>
<td>31.69 ± 0.24</td>
<td>26.25 ± 0.25</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>91.65 ± 0.35</td>
<td>61.52 ± 0.41</td>
<td>45.91 ± 0.41</td>
<td>59.85 ± 0.11</td>
<td>32.06 ± 0.35</td>
<td>24.73 ± 0.22</td>
</tr>
</tbody>
</table>

Table 4: Comparing white-box attack robustness (accuracy %) for ResNet-18 on TinyImageNet. Perturbation size $\epsilon = 8/255$ and step size $\kappa = 1/255$ are used for all methods.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD-20</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>48.79 ± 0.15</td>
<td>23.96 ± 0.15</td>
<td>18.06 ± 0.15</td>
</tr>
<tr>
<td>TRADES</td>
<td>49.11 ± 0.23</td>
<td>22.89 ± 0.26</td>
<td>16.81 ± 0.19</td>
</tr>
<tr>
<td>MART</td>
<td>45.91 ± 0.21</td>
<td>26.03 ± 0.36</td>
<td>19.23 ± 0.23</td>
</tr>
<tr>
<td>GAIRAT</td>
<td>46.09 ± 0.14</td>
<td>17.21 ± 0.33</td>
<td>12.92 ± 0.23</td>
</tr>
<tr>
<td>MAIL-AT</td>
<td>49.72 ± 0.16</td>
<td>24.32 ± 0.31</td>
<td>17.61 ± 0.35</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>51.17 ± 0.19</td>
<td>25.82 ± 0.13</td>
<td>18.68 ± 0.19</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>49.09 ± 0.25</td>
<td>26.65 ± 0.32</td>
<td>18.42 ± 0.21</td>
</tr>
</tbody>
</table>

Comparison with prominent reweighting methods. The results in Tables 1-4 show that, in general, the proposed VIR method is more effective than the two prominent reweighting methods MAIL-AT and...
Table 5: Comparing black-box attack robustness (accuracy %) for Wideresnet-34-10 trained on CIFAR-10.

<table>
<thead>
<tr>
<th>Defense</th>
<th>SQUARE</th>
<th>SPSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>60.12±0.25</td>
<td>61.05±0.16</td>
</tr>
<tr>
<td>TRADES</td>
<td>59.18±0.10</td>
<td>61.15±0.09</td>
</tr>
<tr>
<td>MART</td>
<td>58.72±0.18</td>
<td>58.93±0.11</td>
</tr>
<tr>
<td>GAIRAT</td>
<td>51.97±0.10</td>
<td>52.15±0.30</td>
</tr>
<tr>
<td>MAIL-AT</td>
<td>58.34±0.16</td>
<td>59.24±0.32</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>59.73±0.08</td>
<td>61.45±0.26</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>60.51±0.11</td>
<td>62.59±0.21</td>
</tr>
</tbody>
</table>

GAIRAT, especially under stronger attacks FMN, CW, and Autoattack. For example, VIR-AT significantly outperforms MAIL-AT against FMN-100 (+2.5%), CW (+ 5.0%), and Autoattack (+ 5.0%) attacks on CIFAR-10 using WRN-34-10. The proposed VIR-AT method achieves large improvement margins over GAIRAT against FMN-100 (+ 9.6%), CW (+ 11.1%), and Autoattack (+ 9.6%) on CIFAR-10 using WRN-34-10. VIR-AT also consistently performs better than MAIL-AT and GAIRAT on CIFAR-100 and Tiny-ImageNet against PGD and Autoattack attacks. On SVHN, MAIL-AT slightly outperforms our VIR-AT method against PGD-100, however, VIR-AT performs significantly better against Autoattack by over 4%.

Experimental results presented in Table 5 show that VIR-AT performs better than GAIRAT and MAIL-AT against strong query-based black-box attacks SQUARE and SPSA. Furthermore, the poorer performance recorded by GAIRAT on black-box attacks compared to the PGD-100 white-box attack may potentially indicate that GAIRAT encourages gradient obfuscation according to the arguments made in [Athalye et al., 2018].

Comparison with non-reweighted variants. We applied the proposed reweighting method to a prominent variant of vanilla AT, TRADES [Zhang et al., 2019]. Experimental results in Tables 1-5 show that VIR-TRADES consistently improves upon TRADES against all attacks, especially stronger attacks CW, FMN, and Autoattack.

In addition, the confusion matrices displayed in Figure 2 clearly shows the improvement in class-wise accuracies of data-sets corresponding to harder classes. We show in Figure 3 the class-weight distribution (the sum of all sample weights in each class). From Figure 3, we observe that our proposed instance-wise reweighting assigns the largest weight to the most vulnerable class ‘3’ and the least weight to the least vulnerable class ‘1’ for both VIR-AT and VIR-TRADES.

Figure 2: Confusion matrix displaying robust accuracies under PGD-100 attack using VIR-AT and VIR-TRADES for ResNet18 on CIFAR-10 dataset.

Finally, these experimental results show that the existing reweighting methods GAIRAT and MAIL improved upon the vanilla AT against FGSM and PGD attacks but performed much worse against stronger attacks CW, FMN, and Autoattack. In contrast, our proposed VIR-AT performed significantly better than AT.
Figure 3: Class-weight distribution (sum of all sample weights in each class) for VIR-AT and VIR-TRADES for ResNet18 during training on CIFAR-10 dataset.

on PGD-100 and FGSM without sacrificing performance against CW, FMN, and Autoattack. A likely explanation for the relatively poor performance of existing reweighting functions on stronger attacks such as Autoattack and CW is that they significantly diminish the influence of less vulnerable examples. While upweighting losses of vulnerable examples can be helpful, over-relying on the vulnerable samples may be detrimental as observed in (Dong et al., 2021). Our proposed VIR function achieved a good balance between weighting vulnerable and less vulnerable examples as shown in the ablation studies presented in the next section.

5.6 Ablation Studies and Impact of Hyperparameters

5.6.1 Ablation Studies

We conduct ablation studies on the proposed weight assignment function using ResNet-18 on CIFAR-10. The training settings are the same as those described in Section 5.1. We study the influence of each reweighting component on the robustness against white-box and black-box attacks. The results are presented in Table 6.

Table 6: Ablation studies on VIR-AT showing the impact of reweighting components proposed in Eq. 5 and 6 on white-box and black-box attack robustness (accuracy %).

<table>
<thead>
<tr>
<th>Reweighting component</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
<th>SQUARE</th>
<th>SPSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_{CE}(f_\theta(x_i'), y_i)$</td>
<td>84.12±0.16</td>
<td>51.58±0.17</td>
<td>51.75±0.23</td>
<td>47.92±0.35</td>
<td>55.32±0.09</td>
<td>56.85±0.02</td>
</tr>
<tr>
<td>$S_s(x_i, y_i) \cdot L_{CE}(f_\theta(x_i'), y_i)$</td>
<td>84.35±0.17</td>
<td>54.50±0.11</td>
<td>49.61±0.19</td>
<td>45.48±0.27</td>
<td>54.68±0.15</td>
<td>56.08±0.21</td>
</tr>
<tr>
<td>$S_d(x_i, x_i') \cdot L_{CE}(f_\theta(x_i'), y_i)$</td>
<td>83.90±0.16</td>
<td>51.17±0.14</td>
<td>51.90±0.35</td>
<td>47.95±0.29</td>
<td>56.22±0.19</td>
<td>56.25±0.21</td>
</tr>
<tr>
<td>$w(x_i, x_i', y_i) \cdot L_{CE}(f_\theta(x_i'), y_i)$</td>
<td><strong>84.59±0.18</strong></td>
<td><strong>56.42±0.18</strong></td>
<td><strong>52.18±0.15</strong></td>
<td><strong>48.21±0.08</strong></td>
<td><strong>56.89±0.19</strong></td>
<td><strong>57.35±0.15</strong></td>
</tr>
</tbody>
</table>

Reweighting $L_{CE}(f_\theta(x_i'), y_i)$ with only $S_s(x_i, y_i)$ improves robustness against PGD-100, but yields reduced robustness against CW and Autoattack. When $L_{CE}(f_\theta(x_i'), y_i)$ is reweighted using $S_d(x_i, x_i')$, no significant improvement in robustness was observed against PGD-100, however, stable performance over stronger attacks are observed. Reweighting $L_{CE}(f_\theta(x_i'), y_i)$ with the proposed reweighting function significantly improves robustness accuracy against both white-box and black-box attacks.

The ablation studies on VIR-TRADES on CIFAR-10 using ResNet-18 are presented in Table 7.
In this paper, we propose a novel vulnerability-aware instance-wise reweighting strategy for adversarial training. The proposed reweighting strategy takes into consideration the intrinsic vulnerability of natural examples used for crafting adversarial examples during adversarial training. We show that existing reweighting methods fail to achieve significant robustness against stronger white-box and black-box attacks. Lastly, we experimentally show that the proposed reweighting function effectively improves adversarial training without diminishing its performance against stronger attacks. In addition, the proposed method shows improvement on every dataset evaluated.

5.6.2 Impact of Hyperparameters

We provide a brief discussion of the impact of the hyperparameters $\gamma$, $\alpha$, and $\beta$ used in the proposed reweighting function. Hyperparameter $\gamma$ appears in the power of the exponential function in Eq. 5 Setting $\gamma$ high results in a large disparity in weights between vulnerable examples and less vulnerable examples. $\alpha$ helps with the numerical stability of Eq. 5 Introducing $\beta$ into the reweighting function in Eq. 7 allows for a balance between relative weights of vulnerable examples and less vulnerable examples. Without adding $\beta$, the reweighting function could potentially return very low values, which significantly diminishes the influence of less vulnerable training samples. On the other hand, if $\beta$ is set too high, the desired reweighting effect is lost. The values of these hyperparameters are heuristically determined. Experimental studies on the impact of the hyperparameters on the performance are shown in Appendix C.

6 Conclusion

In this paper, we propose a novel vulnerability-aware instance-wise reweighting strategy for adversarial training. The proposed reweighting strategy takes into consideration the intrinsic vulnerability of natural examples used for crafting adversarial examples during adversarial training. We show that existing reweighting methods fail to achieve significant robustness against stronger white-box and black-box attacks. Lastly, we experimentally show that the proposed reweighting function effectively improves adversarial training without diminishing its performance against stronger attacks. In addition, the proposed method shows improvement on every dataset evaluated.

References


Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. Advances in neural information processing systems, 32, 2019.


Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.


Han Xu, Xiaorui Liu, Yaxin Li, Anil Jain, and Jiliang Tang. To be robust or to be fair: Towards fairness in adversarial training. In International Conference on Machine Learning, pp. 11492–11501. PMLR, 2021.


A Appendix: Theoretical Explanation of Input Vulnerability

In this section, we show that, under a special setting, the example vulnerability directly corresponds to the probability of predicting the true class $y$. We consider a binary classification task of natural examples sampled from a Gaussian mixture distribution, following the settings described in (Carmon et al., 2019; Schmidt et al., 2018; Xu et al., 2021; Ma et al., 2022). We formally define the settings below.

**Definition A.1 (Gaussian Mixture Distribution.)** Let $-\mu$, $\mu$ be the mean parameters corresponding to data sampled from two classes $y = \{-1, +1\}$ according to Gaussian distribution.

Let $\sigma_-\!, \sigma_+$ represent the variances of the classes $-1$ and $+1$. Then the Gaussian mixture model is defined by the following distribution over $(x, y) \in \mathbb{R}^d \times \{\pm 1\}$:

$$
\begin{align*}
  y &\sim \{-1, +1\} \quad \mu = (\eta, \ldots, \eta) \\
  x &\sim \begin{cases} 
    \mathcal{N}(\mu, \sigma_+^2 \mathcal{I}), & \text{if } y = +1 \\
    \mathcal{N}(\mu, \sigma_-^2 \mathcal{I}), & \text{if } y = -1 
  \end{cases}
\end{align*}
$$

$I$ is the $d$-dimension identity matrix.

To design classes with different intrinsic vulnerabilities, we ensure that there is a K-factor difference between the variances of the classes such that $\sigma_-, \sigma_+ = 1 : K$ and $K > 1$. The larger variance of the examples corresponding to class $+1$ intuitively suggests higher vulnerability than an example corresponding to class $-1$. 

15
The analysis is done using a linear classifier as follows:
\[ f(\mathbf{x}) = \text{sign}(\langle \omega, \mathbf{x} \rangle + b) \] (12)
where \( \omega \) and \( b \) respectively represent the model weights and bias term, and \( \text{sign}(z) \) returns 1 if \( z \geq 0 \), else -1. The natural risk is denoted as:
\[
\begin{align*}
R_{\text{nat}}^- &= \Pr(\mathcal{N}(0,1) \leq A - K \cdot \sqrt{A^2 + q(K)}) \\
R_{\text{nat}}^+ &= \Pr(\mathcal{N}(0,1) \leq -K \cdot A + \sqrt{A^2 + q(K)})
\end{align*}
\] (13)
where \( A = \frac{2}{K - 1} \cdot \sqrt{b} \) and \( q(K) = \frac{2 \log K}{K^2 - 1} \) which is a positive constant and depends only on \( K \). Therefore, class +1 has a larger risk: \( R_{\text{nat}}^-(f_{\text{nat}}) < R_{\text{nat}}^+(f_{\text{nat}}) \).

Theorem 1 shows that the vulnerable class +1 (with larger variance) is more difficult to classify than -1, because the optimal \( f_{\text{nat}} \) has a higher standard error for +1 than -1.

Corollary 1 (Vulnerability of a data sample.) The vulnerability of an example may be estimated using a classifier’s estimated class-probability.

Proof 1 The class-wise risks corresponding to classes -1 and +1 can be respectively written as:
\[
\begin{align*}
R_{\text{nat}}^-(f_{\text{nat}}) &= \Pr(f_{\text{nat}}(\mathbf{x}) = -1|y = +1) = \Pr(\langle \omega, \mathbf{x} \rangle + b > 0) \\
R_{\text{nat}}^+(f_{\text{nat}}) &= \Pr(f_{\text{nat}}(\mathbf{x}) = +1|y = -1) = \Pr(\langle \omega, \mathbf{x} \rangle + b < 0)
\end{align*}
\]
Let \( P_{\text{nat}}^- \) and \( P_{\text{nat}}^+ \) respectively denote the correct probability estimates of classes -1 and +1.

Then,
\[
\begin{align*}
P_{\text{nat}}^-(f_{\text{nat}}) &= \Pr(f_{\text{nat}}(\mathbf{x}) = -1|y = -1) \\
P_{\text{nat}}^+(f_{\text{nat}}) &= \Pr(f_{\text{nat}}(\mathbf{x}) = +1|y = +1).
\end{align*}
\]
From Theorem 1, \( R_{\text{nat}}^-(f_{\text{nat}}) < R_{\text{nat}}^+(f_{\text{nat}}) \). It follows that \( P_{\text{nat}}^-(f_{\text{nat}}) > P_{\text{nat}}^+(f_{\text{nat}}) \).

B Additional Experiments on Data Augmentation Based Defense

We perform additional experiments on the data augmentation-based defense using extra datasets generated using the Elucidating Diffusion Model (EDM) following Wang et al. [2023].

We utilize Wideresnet-28-10 as the backbone model for the experiments. The model was trained using 20M EDM-generated data for 400 epochs with the training batch size of 512. Common augmentation described in (Wang et al. 2023) is applied on the training samples. We employ the SGD optimizer with Nesterov momentum ([Nesterov 1983] where the momentum factor and weight decay were set to 0.9 and 5e^{-4} respectively. We use the cyclic learning rate schedule with cosine annealing ([Smith & Topin 2019], where the initial learning rate is set to 0.2. For VIR-TRADES, we used the settings described in section 5.3, while for VIR-AT, we set \( \gamma \) to 5.0, \( \beta \) to 0.2 and \( \alpha \) to 7.0.

The obtained results are reported in Table 8. The experimental results show improvement of VIR-TRADES over TRADES and VIR-AT over AT on CIFAR-10.
Table 8: Comparing robustness (accuracy %) for Wideresnet-28-10 on CIFAR-10 trained on 20M EDM-generated data.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRADES</td>
<td>90.33 ± 0.14</td>
<td>65.74 ± 0.16</td>
<td>64.01 ± 0.17</td>
<td>63.03 ± 0.13</td>
</tr>
<tr>
<td>AT</td>
<td>91.56 ± 0.09</td>
<td>66.23 ± 0.12</td>
<td>65.62 ± 0.09</td>
<td>62.42 ± 0.11</td>
</tr>
<tr>
<td>VIR-TRADES</td>
<td>89.73 ± 0.12</td>
<td>67.23 ± 0.08</td>
<td>65.47 ± 0.05</td>
<td>64.31 ± 0.17</td>
</tr>
<tr>
<td>VIR-AT</td>
<td>92.69 ± 0.25</td>
<td>67.98 ± 0.25</td>
<td>66.95 ± 0.17</td>
<td>62.95 ± 0.09</td>
</tr>
</tbody>
</table>

C Ablation Studies on Hyperparameters

We show in the following experimental results on varying hyperparameter $\alpha, \gamma, \beta$ values. The results are obtained from training ResNet-18 on CIFAR-10 dataset.

Table 9: Ablation studies on VIR-AT showing the impact of the $\beta$ hyperparameter on the performance of the proposed reweighting function.

<table>
<thead>
<tr>
<th>Reweighting Function</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 0)$</td>
<td>84.48 ± 0.11</td>
<td>57.20 ± 0.14</td>
<td>51.25 ± 0.12</td>
<td>47.32 ± 0.11</td>
</tr>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 0.007)$</td>
<td>84.59 ± 0.15</td>
<td>56.42 ± 0.18</td>
<td>52.18 ± 0.15</td>
<td>48.21 ± 0.08</td>
</tr>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 0.1)$</td>
<td>84.26 ± 0.07</td>
<td>53.52 ± 0.13</td>
<td>52.20 ± 0.19</td>
<td>48.17 ± 0.13</td>
</tr>
</tbody>
</table>

Table 10: Ablation studies on VIR-TRADES showing the impact of the $\beta$ hyperparameter on the performance of the proposed reweighting function.

<table>
<thead>
<tr>
<th>Reweighting Function</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 0.5)$</td>
<td>83.32 ± 0.18</td>
<td>53.70 ± 0.15</td>
<td>51.60 ± 0.12</td>
<td>49.29 ± 0.09</td>
</tr>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 1.6)$</td>
<td>82.03 ± 0.13</td>
<td>54.86 ± 0.17</td>
<td>53.11 ± 0.17</td>
<td>51.03 ± 0.16</td>
</tr>
<tr>
<td>$S_v(x_i, y_i) \cdot S_d(x_i, x'_i) + (\beta = 2.0)$</td>
<td>80.86 ± 0.10</td>
<td>54.58 ± 0.15</td>
<td>52.62 ± 0.08</td>
<td>50.56 ± 0.13</td>
</tr>
</tbody>
</table>
Table 11: Ablation studies on VIR-AT showing the impact of the $\alpha$ and $\gamma$ hyperparameters on the performance of the proposed reweighting function.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\gamma$</th>
<th>$\beta$</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>0.007</td>
<td>83.75±0.11</td>
<td>54.38±0.08</td>
<td>52.20±0.15</td>
<td>48.25±0.15</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>0.007</td>
<td>83.80±0.09</td>
<td>55.28±0.05</td>
<td>52.13±0.15</td>
<td>48.18±0.16</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>0.007</td>
<td>84.16±0.06</td>
<td>55.85±0.10</td>
<td>52.05±0.11</td>
<td>48.15±0.11</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>0.007</td>
<td>84.31±0.06</td>
<td>55.92±0.10</td>
<td>51.80±0.06</td>
<td>48.25±0.09</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>0.007</td>
<td>84.59±0.18</td>
<td>56.42±0.18</td>
<td>52.20±0.10</td>
<td>48.17±0.13</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>0.007</td>
<td>84.19±0.07</td>
<td>56.68±0.12</td>
<td>52.01±0.06</td>
<td>48.02±0.10</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>0.007</td>
<td>83.96±0.15</td>
<td>54.75±0.13</td>
<td>51.90±0.08</td>
<td>47.89±0.05</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>0.007</td>
<td>84.52±0.08</td>
<td>54.92±0.10</td>
<td>52.01±0.10</td>
<td>47.97±0.06</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0.007</td>
<td>84.36±0.08</td>
<td>55.82±0.11</td>
<td>51.76±0.08</td>
<td>47.70±0.08</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>0.007</td>
<td>84.15±0.08</td>
<td>56.30±0.11</td>
<td>51.91±0.12</td>
<td>47.87±0.07</td>
</tr>
</tbody>
</table>

Table 12: Ablation studies on VIR-TRADES showing the impact of the $\alpha$ and $\gamma$ hyperparameters on the performance of the proposed reweighting function.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\gamma$</th>
<th>$\beta$</th>
<th>Natural</th>
<th>PGD-100</th>
<th>CW</th>
<th>AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.0</td>
<td>1.6</td>
<td>82.10±0.15</td>
<td>54.07±0.08</td>
<td>52.29±0.09</td>
<td>50.08±0.12</td>
</tr>
<tr>
<td>2</td>
<td>3.0</td>
<td>1.6</td>
<td>82.13±0.11</td>
<td>54.10±0.05</td>
<td>52.34±0.09</td>
<td>50.13±0.08</td>
</tr>
<tr>
<td>3</td>
<td>3.0</td>
<td>1.6</td>
<td>82.12±0.11</td>
<td>54.16±0.10</td>
<td>52.51±0.10</td>
<td>50.41±0.07</td>
</tr>
<tr>
<td>5</td>
<td>3.0</td>
<td>1.6</td>
<td>82.06±0.11</td>
<td>54.31±0.08</td>
<td>52.54±0.13</td>
<td>50.43±0.05</td>
</tr>
<tr>
<td>7</td>
<td>3.0</td>
<td>1.6</td>
<td>82.05±0.10</td>
<td>54.52±0.09</td>
<td>52.59±0.11</td>
<td>50.52±0.05</td>
</tr>
<tr>
<td>8</td>
<td>3.0</td>
<td>1.6</td>
<td>82.03±0.13</td>
<td>54.86±0.17</td>
<td>53.11±0.17</td>
<td>51.03±0.16</td>
</tr>
<tr>
<td>8</td>
<td>1.0</td>
<td>1.6</td>
<td>81.05±0.10</td>
<td>53.96±0.13</td>
<td>51.85±0.06</td>
<td>49.89±0.09</td>
</tr>
<tr>
<td>8</td>
<td>2.0</td>
<td>1.6</td>
<td>81.09±0.12</td>
<td>54.26±0.15</td>
<td>52.41±0.08</td>
<td>50.28±0.17</td>
</tr>
<tr>
<td>8</td>
<td>4.0</td>
<td>1.6</td>
<td>81.91±0.09</td>
<td>54.42±0.13</td>
<td>52.70±0.07</td>
<td>50.34±0.11</td>
</tr>
</tbody>
</table>